

On the Connectedness of Clash-free Timetables*

Moritz Mühlenthaler[†]

Rolf Wanka

July 13, 2015

Abstract

We investigate the connectedness of clash-free timetables with respect to the Kempe-exchange operation. This investigation is related to the connectedness of the search space of timetabling problem instances, which is a desirable property, for example for two-step algorithms using the Kempe-exchange during the optimization step. The theoretical framework for our investigations is based on the study of reconfiguration graphs, which model the search space of timetabling problems. We contribute to this framework by including timeslot availability requirements in the analysis and we derive improved conditions for the connectedness of clash-free timetables in this setting. We apply the theoretical insights to establish the connectedness of clash-free timetables for a number of benchmark instances.

1 Introduction

Timetabling problems in the context of a university ask for an assignment of events (e.g., courses or exams) to rooms and timeslots such that no two conflicting events are scheduled simultaneously. By a straightforward reduction from vertex coloring it is immediate that such timetabling problems are NP-hard, which motivates the use of (meta-)heuristics in order to solve timetabling problems in practice; see for example [32] for an overview of different problem models and solution approaches. According to the classification of heuristic optimization algorithms for timetabling problems in [21], many approaches in the literature fall in the category of *two-step optimization algorithms*. The general procedure is the following: In the first step, the underlying search problem is solved and the resulting feasible solution is used as a starting point for the second step, during which the optimization is performed. In the second step only feasible solutions are considered. A recent example of a state-of-the-art two-step approach is [23], numerous other examples can be found in [21]. During the optimization step, feasible timetables are modified using Kempe-exchanges or similar operations that preserve their feasibility. It is natural to ask whether any feasible timetable, in particular an optimal one, can be reached from an initial feasible timetable. We give a partial answer to this question by investigating conditions that establish the connectedness of the search space of *clash-free* timetables.

A timetable is clash-free, if no two conflicting events are scheduled simultaneously. We model the structure of the search space of clash-free timetables in terms of reconfiguration graphs. Such graphs arise in the context of reconfiguration problems: Given an instance \mathcal{I} of a combinatorial search problem, the corresponding *reconfiguration problem* asks whether one feasible solution to \mathcal{I} can be transformed into another feasible solution in a step-by-step manner by making local changes, such that each intermediate solution is also feasible. A *reconfiguration graph* has as nodes the feasible solutions of the underlying combinatorial problem and two such solutions are adjacent whenever there is a local change that transforms one into the other. Reconfiguration variants of classical combinatorial problems and their reconfiguration graphs have been studied in the literature [4, 16, 17, 19, 18, see e.g.]. The heart of the matter of timetabling problems in the academic context (in contrast to the “high school timetabling” model, see [9, Section 2], [29]) is the vertex coloring problem: A clash-free timetable corresponds to a proper coloring of the event conflict graph, see e.g. [9]. [8] have shown that determining the connectedness of any two proper colorings of a graph is PSPACE-complete for four or more colors and tractable otherwise, in a setting where an admissible local change alters the color of a single vertex. A related line

*This is the extended version of [28] presented at PATAT 2014.

[†]Research funded in parts by the School of Engineering of the University of Erlangen-Nuremberg.

of research deals with the question whether two given proper colorings are connected, see e.g., [5, 35]. [20] give sufficient (but not necessary) conditions for the connectedness of any two vertex colorings with respect to the Kempe-exchange operation. The Kempe-exchange is a generalization of the local change mentioned above. It is a popular operation used by algorithms for timetabling problems for exploring the search space, including many of the two-step algorithms in the references above. Therefore, the results of [20] are applicable in the timetabling context, see Corollary 1. Basically, the clash-free timetables are connected if the number of timeslots is sufficiently large compared to the degeneracy of the graph of event conflicts. Fortunately, this condition can be checked efficiently.

Clash-freeness is typically not the only requirement for a timetable to be feasible. In many problem formulations in research and practice [10, 7, 33, 6, e.g.], certain timeslots or rooms may be unavailable/unsuitable for particular events and it is required that each event is placed strictly in the available rooms and timeslots. We employ a standard reduction from list to vertex coloring in order to include timeslot availability requirements in the reconfiguration model. We show that this approach leads to a faithful representation of the search space with respect to its connectedness and its diameter. It turns out that due to the nature of the reduction the condition in Corollary 1 is too strict to be useful for certifying the connectedness of the clash-free timetables that satisfy the timeslot availability requirements. However, we extend the techniques from [20] to derive improved conditions in this setting. For this purpose, we introduce the *subdegeneracy* of a graph, which generalizes the notion of degeneracy by ignoring the potential contribution to the degeneracy of a given subgraph. We show that the clash-free timetables that satisfy the timeslot availability requirements are connected with respect to the Kempe-exchange if there are sufficiently many timeslots compared to the subdegeneracy of the conflict graph and a suitably chosen subgraph. In contrast to the degeneracy, which can be computed in linear time, the computational complexity of determining the subdegeneracy is an open problem and we propose a heuristic solution approach. We further provide data on the connectedness of the clash-free timetables for a number of benchmarking instance sets, including artificial and real-world instance, with and without taking timeslot availability requirements into account.

The remainder of this work is organized as follows: In Section 2 we provide the basic formalisms required for our analysis of the connectedness of clash-free timetables presented in Section 3. In Section 4 we investigate the connectedness of the clash-free timetables for number of standard benchmarking instance sets.

2 Background

2.1 The University Timetabling Problem

The University Timetabling Problem (UTP) formalizes in terms of a search problem the task of creating a course or examination schedule at a university.

Definition 1 (University Timetabling Problem (UTP)).

INSTANCE:

- a set of events $E = \{e_1, \dots, e_n\}$
- a set of rooms $R = \{r_1, \dots, r_\ell\}$
- a set of timeslots $P = \{p_1, \dots, p_k\}$
- a graph $G = (E, L)$ with nodes E and edges $L \subseteq \{\{u, v\} \mid u, v \in E\}$

The graph G is referred to as the *conflict graph*. Two events are called *conflicting* if they are adjacent in G . An element of the set $P \times R$ is referred to as *resource*. A *timetable* τ is an assignment $\tau : E \rightarrow P \times R$. Two events e, e' are *overlapping*, if $e \neq e'$ and $\tau(e) = \tau(e')$. A timetable is called *overlap-free* if no two events overlap. Two events e, e' are *clashing* in τ , if they are conflicting and they are assigned to the same timeslot. A timetable is *feasible*, if it is clash-free and overlap-free.

TASK: Find a feasible timetable.

It is usually assumed that all timeslots have the same length and that each event fits in a single timeslot. The UTP as defined above is equivalent to the problem given in [9, Section 3.4] and generalizes

many of the more refined problem formulations in the literature [6, 12, 26, see e.g.]. The clash-freeness requirement and its relation to the vertex coloring problem is the heart of the matter of timetabling problems in the academic context [9, 32, see e.g.]. Other kinds of requirements such as *availability requirements* and *precedence requirements* often occur in practice [7, 33, e.g.] and in the benchmarking problem models [6, 12, 26, e.g.]. Later, we will consider the UTP above with additional timeslot availability requirements. These requirements mandate that only specific timeslots can be assigned to an event. We formalize timeslot availability requirements in terms of an availability function α , which determines for each event the set of available timeslots

$$\alpha : E \rightarrow \mathcal{P}(P) .$$

An important subproblem of the UTP is the *room assignment problem*. Given a timeslot $p \in P$, then events $E' \subseteq E$ admit a room assignment, if there is an assignment $\rho : E' \rightarrow R$ such that $(p, \rho(e))$ is available for each $e \in E'$.

2.2 Vertex Coloring

A graph $G = (V(G), E(G))$, for short $G = (V, E)$, consists of a set of *vertices* V and a set of *edges* $E \subseteq \{\{u, v\} \mid u, v \in V\}$. Unless stated otherwise, we assume that graphs are loopless and finite. We denote by $u - v$ that the vertices u and v are adjacent, i.e., $\{u, v\} \in E$. The graph $G[U]$ denotes the subgraph of G induced by the vertices $U \subseteq V(G)$. A *(vertex)- k -coloring* of a graph G is a mapping $c : V \rightarrow \{1, \dots, k\}$ that assigns one of the colors $\{1, \dots, k\}$ to each vertex of G . A coloring is called *proper*, if no two adjacent nodes have the same color. Unless stated otherwise, we will use the term *coloring* as a shorthand for *proper coloring*. The *vertex k -coloring problem* asks, whether a graph admits a k -coloring. A k -coloring c of G partitions the vertices of G into k sets of independent (mutually non-adjacent) vertices called *color classes*. A color class $a \in \{1, \dots, k\}$ contains all vertices of color a . We denote by $G(a, b)$ the bipartite subgraph induced by the color classes a and b . A connected component in $G(a, b)$ is referred to as *Kempe-component*.

Given a set $L(v)$ (called *list*) of available colors for each $v \in V$, a *list coloring* $c : V \rightarrow \bigcup_{v \in V} L(v)$ of G is a coloring of G such that $c(v) \in L(v)$ for each $v \in V$. Vertex coloring is a special case of list coloring, where all colors are available for each node. By using a standard technique [9, Proposition 3.2] list coloring can be reduced to vertex coloring: Let the colors be labeled $1, \dots, k$, where $k = |\bigcup_{v \in V} L(v)|$. Now, let the graph H_G be a copy of G to which we add a clique C on k (new) nodes v_1, \dots, v_k . For each $v \in V(G)$, we add an edge $v - v_i$ to H_G , whenever $i \notin L(v)$. Clearly, H_G admits a k -coloring if and only if G admits a list coloring. The problem of deciding if a given UTP instance admits a clash-free timetable that satisfies timeslot availability requirements is equivalent to deciding if the conflict graph admits a list coloring, where the $L(e) = \alpha(e)$ for each event e .

2.3 The Vertex Coloring Reconfiguration Problem

Reconfiguration problems formalize the question, if a solution to a problem instance can be transformed into another solution in a step-by-step manner by some reconfiguration operation, such that each intermediate solution is feasible [17]. To show that a search space is connected we need to check whether *any two* solutions are connected. In the context of the vertex coloring problem this question has been investigated for example in [27, 8, 5, 3, 14]. As a reconfiguration operation, elementary recolorings and Kempe-exchanges have been considered in the literature. Given a coloring c of a graph G , an *elementary recoloring* changes the color of a single vertex u of G to a color that does not occur in the neighborhood of u . Two k -colorings c_1 and c_2 of G are adjacent, $c_1 \sim_E c_2$, if there is an elementary recoloring that transforms c_1 into c_2 . The Kempe-exchange is a generalization of the elementary recoloring operation. Given two colors a and b , a Kempe-exchange switches the colors of a Kempe-component, i.e., a connected component in $G(a, b)$. The result of this operation is a new coloring, such that, within the Kempe-component, each vertex of the of color a is assigned to color b and vice versa. An elementary recoloring that changes the color of a vertex u from a to b is a Kempe-exchange on the Kempe-component containing u in $G(a, b)$, which is an isolated vertex. Two colorings c_1 and c_2 of G are adjacent with respect to the Kempe-exchange, denoted by $c_1 \sim_K c_2$, if there is a Kempe-exchange that transforms c_1 into c_2 . Each of the two adjacency relations \sim_E and \sim_K gives rise to a graph structure on the set of k -colorings of G .

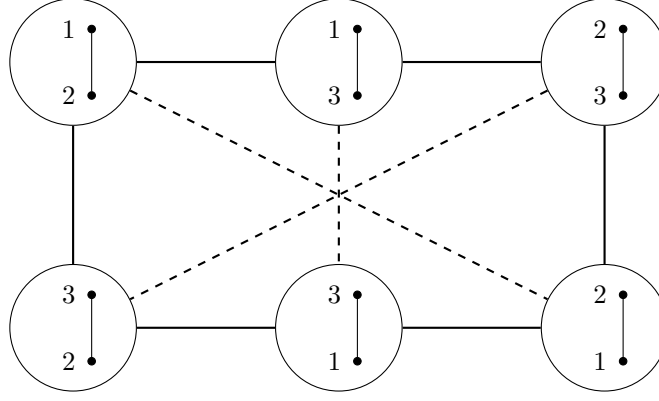


Figure 1: The Kempe-3-coloring graph $\mathcal{K}_3(K_2)$ of the graph K_2 . The subgraph induced by the solid edges corresponds to $\mathcal{C}_3(K_2)$. Each dashed edge represents a Kempe-exchange that cannot be realized by a single elementary recoloring.

Definition 2 ((Kempe-) k -coloring graph). For a graph $G = (V, E)$ and $k \in \mathbb{N}$ let

$$\begin{aligned}\mathcal{V} &:= \{c : V \rightarrow \{1, \dots, k\} \mid c \text{ is a } k\text{-coloring of } G\} \\ \mathcal{E}_E &:= \{\{c_1, c_2\} \mid c_1, c_2 \in \mathcal{V} \text{ and } c_1 \sim_E c_2\} \\ \mathcal{E}_K &:= \{\{c_1, c_2\} \mid c_1, c_2 \in \mathcal{V} \text{ and } c_1 \sim_K c_2\} .\end{aligned}$$

Then the k -coloring graph is the graph $\mathcal{C}_k(G) = (\mathcal{V}, \mathcal{E}_E)$. The Kempe- k -coloring graph is the graph $\mathcal{K}_k(G) = (\mathcal{V}, \mathcal{E}_K)$.

Algorithm 1: KEMPERECONFIGURATION

input : graph G , labeling v_1, \dots, v_n of the vertices, k -colorings c_1, c_2 of G
output: list of Kempe-exchanges transforming c_1 into c_2
data : array c of length n storing the current color of each vertex, list K of Kempe-exchanges
 $K \leftarrow$ empty list
for $i \leftarrow 1$ **to** n **do**
 $H \leftarrow G[v_1, \dots, v_i]$
 for $i \leftarrow 1$ **to** n **do**
 $c[i] \leftarrow c_1(v_i)$
 /* Kempe-exchange $\kappa = (a, b, u)$, where a, b are colors and $u \in V(H)$ */
 for $\kappa = (a, b, u) \in K$ **do**
 without loss of generality let $a \neq c[i]$
 if $c[i] = b$ and v_i has exactly one neighbor of color a in H **then**
 /* Note that the color of v_i will be changed by κ in H */
 if $c[i] = b$ and v_i has at least two neighbors of color a in H **then**
 choose color $b' \neq b$, which is not used by any neighbor of v_i in H
 insert Kempe-exchange $k = (b, b', v_i)$ right before κ in K and apply k to c
 apply Kempe-exchange κ to c
 append Kempe-exchange $(c_2(v_i), c[i], v_i)$ to K
return K

Figure 1 shows as a toy example $\mathcal{C}_3(K_2)$ and $\mathcal{K}_3(K_2)$, where K_2 is the complete two-vertex graph. Clearly, for any graph G and $k \geq 1$, $\mathcal{C}_k(G) \subseteq \mathcal{K}_k(G)$. The diameter and the connectedness of (Kempe-) k -coloring graphs has been investigated for example in [27, 3, 14]. To the best of our knowledge, in

general graphs and for $k \geq 4$, the complexity of deciding the connectedness of any two k -colorings of a graph is still open. However, in [20], a sufficient (but not necessary) condition for the connectedness of any two k -colorings is given, which relates the connectedness of the colorings to the degeneracy of the graph to be colored. A graph G is called k -degenerate, if its vertices can be linearly ordered such that each vertex has at most k neighbors preceding it. The smallest k for which G admits such an ordering is the *degeneracy* $\deg(G)$, which is sometimes also called *width* of G . A witness vertex ordering of $\deg(G)$ can be found in linear time by repeatedly removing vertices of minimal degree [25, 34, 2]. Equivalently, $\deg(G)$ is the largest minimum degree of any subgraph of G . Let $S(G)$ be the set of orderings of the vertices of G and let $\text{pred}(v, \sigma)$ denote the number of neighbors of the vertex $v \in V(G)$ that precede v in the ordering $\sigma \in S(G)$. In formal terms, the two characterizations of $\deg(G)$ can be stated as follows:

$$\deg(G) := \max_{H \subseteq G} \min_{v \in V(H)} \{d_H(v)\} = \min_{\sigma \in S(G)} \max_{v \in V(G)} \text{pred}(v, \sigma) , \quad (1)$$

where $d_H(v)$ denotes the degree of v in H . The degeneracy of a graph is an upper bound on its chromatic number. In [20], degeneracy has been used in order to establish the connectedness of Kempe- k -coloring graphs as follows:

Theorem 1 ([20, Proposition 2.1]). For any graph G , the Kempe- k -coloring graph $\mathcal{K}_k(G)$ is connected if $k > \deg(G)$.

The proofs given in [20] and [27] are essentially an analysis of the algorithm KEMPERECONFIGURATION shown in Algorithm 1. This algorithm transforms a source coloring c_1 into a destination coloring c_2 by a sequence of Kempe-exchanges, provided that a sufficient number of colors is available. The vertices are processed one-by-one according to the given labeling. The idea behind the algorithm is to prevent that changing the color of the current vertex interferes with colors of the previously processed vertices.

3 The Connectedness of Clash-free Timetables

We investigate the connectedness of the search space of clash-free timetables with respect to the Kempe-exchange operation. In the following, let G be the conflict graph of a UTP instance \mathcal{I} with timeslots $\{1, \dots, p\}$. We consider timetables that differ only with respect to the room assignment to be equivalent. Therefore, each p -coloring of G corresponds to an equivalence class of clash-free timetables and the adjacency relation \sim_K on the p -colorings of G induces an adjacency relation on the equivalence classes of clash-free timetables. As a consequence, $\mathcal{K}_p(G)$ represents the search space of clash-free timetables of clash-free timetables connected by Kempe-exchanges. If $\mathcal{K}_p(G)$ is connected, then a two-step algorithm using Kempe-exchanges for search space exploration can reach an optimal solution from any starting point. Otherwise, the algorithm may fail to find an optimal solution due to the structure of the search space.

A sufficient condition establishing the connectedness of clash-free timetables result directly from Theorem 1:

Corollary 1. The search space of clash-free timetables is connected if $p > \deg(G)$.

In most applications however, clash-freeness is not the only requirement a timetable needs to satisfy. In addition, timeslot availability requirements, room availability requirements, and overlap-freeness requirements may restrict the set of feasible timetables, and, as a consequence, limit the search space to a certain subgraph of $\mathcal{K}_p(G)$. In particular, for the additional requirements above, the search space is restricted to the following nodes of $\mathcal{K}_p(G)$:

1. timeslot availability requirements:

$$C_\pi = \{c \in V(\mathcal{K}_p(G)) \mid \forall v \in V(G) : c(v) \text{ is available for event } v\}$$

2. overlap-freeness and room availability requirements:

$$C_\rho = \{c \in V(\mathcal{K}_p(G)) \mid \forall i \in P : \text{color class } i \text{ admits a room assignment}\}$$

Regarding overlap-freeness and room availability requirements, to the best of our knowledge, the properties of the corresponding reconfiguration graphs have not been studied so far. The *bounded vertex k -coloring problem* with bound $b \in \mathbb{N}$ is the problem of coloring a graph with k colors such that the size of each color class is at most b . The bounded vertex coloring problem has been studied for example in [24] as well as [1] in the setting of unit-time task scheduling on multiple processors, and in [11] in the timetabling context. If overlap-freeness is required and no particular room availability requirements are present, then the graph $\mathcal{K}_P(G)[C_\rho]$ is the reconfiguration graph of a bounded vertex coloring instance. The reconfiguration variant of the bounded vertex coloring problem seems to be an interesting problem which deserves further investigation. The situation gets more involved if room availability requirements are present: Checking if the k events in a color class admit a room assignment is equivalent to checking if a suitably chosen bipartite graph admits a matching of cardinality k .

We now investigate conditions that certify the connectedness of $\mathcal{K}_p(G)[C_\pi]$. Using the standard reduction from list coloring to vertex coloring described in Section 2.2, we obtain a graph H_G that contains the original conflict graph G and a clique on p additional vertices v_1, \dots, v_p , which is used for representing the available timeslots for each event. Our goal is to show that the reconfiguration graph of the p -colorings of H_G is a suitable representation of the search space of clash-free timetables that satisfy given timeslot availability requirements. First, we show that $\mathcal{K}_p(H_G)$ is connected if and only if $\mathcal{K}_p(G)[C_\pi]$ is connected. Please note that, due to the nature of the reduction, there are Kempe-exchanges on p -colorings of H_G for which there is no corresponding Kempe-exchange on G ; just consider Kempe-exchanges that involve the nodes v_1, \dots, v_p . We give further evidence that $\mathcal{K}_p(H_G)$ is a suitable representation of the search space $\mathcal{K}_p(G)[C_\pi]$, by showing that their diameters differ only by a factor linear in $|V(G)|$.

3.1 Search space representation in terms of $\mathcal{K}_p(H_G)$

We first show that $\mathcal{K}_p(H_G)$ is connected precisely when $\mathcal{K}_p(G)[C_\pi]$ is connected. For this purpose, we construct from $\mathcal{K}_p(G)[C_\pi]$ an auxiliary graph K . The vertices of K are the vertices of $\mathcal{K}_p(G)[C_\pi]$. Any two vertices $u, v \in V(K)$ (i.e., colorings of G) are adjacent if there is an $u - v$ edge in $\mathcal{K}_p(G)$ or if there are two colors i and j such that u can be transformed into v by swapping the colors in all except a single connected component of $G(i, j)$. For technical reasons that involve the construction of a graph homomorphism to K we add a self-loop to each node of K . From the construction of K it follows easily that K is connected if and only if $\mathcal{K}_p(G)$ is connected:

Proposition 1. K is connected if and only if $\mathcal{K}_p(G)$ is connected.

Proof. The edges which occur in K but not in G are merely shortcuts for several individual Kempe-exchanges performed on G . Therefore, $\mathcal{K}_p(G)[C_\pi]$ is connected if and only if K is connected. \square

Figure 2 shows the various graphs under consideration and how they are related for a small list-coloring instance consisting of a graph $G = (\{u, v\}, \{u - v\})$ and color lists $L(u) = \{1\}$ and $L(v) = \{2\}$. The nodes v_1 and v_2 of H_G were added by the reduction from list coloring to vertex coloring.

Proposition 2. There is a graph homomorphism $f : \mathcal{K}_p(H_G) \rightarrow K$.

Proof. We construct the mapping $f : V(\mathcal{K}_p(H_G)) \rightarrow V(K)$. Let $c \in V(\mathcal{K}_p(H_G))$. First, we swap the colors v_1, \dots, v_p such that v_i has color i for each $i \in \{1, \dots, p\}$. This can be achieved by applying the following sequence of Kempe-exchanges to the coloring c : For each color $j \in \{1, \dots, p\}$, if the current color of v_j is $i \neq j$, swap the colors in each Kempe-component of $H_G(i, j)$. Let c' be the resulting coloring. Except for the vertices v_1, \dots, v_p and their incident edges, H_G is just a copy of $V(G)$. Now, pick $f(c) = \tilde{c}$, where \tilde{c} is equivalent to c' restricted to the vertices $V(G) \subseteq V(H_G)$. Clearly, \tilde{c} is a proper coloring of $\mathcal{K}_p(G)$. Due to the construction of H_G , \tilde{c} satisfies the list coloring requirements for G , i.e., for each $v \in V(G)$ we have $c(v) \in \alpha(v)$. Therefore, $\tilde{c} \in V(K) = V(\mathcal{K}_p(G)[C_\pi])$.

We show that the mapping f is a graph homomorphism as required. Let c, d be colorings of H_G such that $c - d$ in $\mathcal{K}_p(H_G)$. Further, let κ be a witness of $c \sim_K d$. There are two cases to consider:

1. The Kempe-exchange κ does not involve any of the nodes v_1, \dots, v_p . Then f renames the color classes of the colorings c and d if required and there is a Kempe-exchange corresponding to κ that establishes $f(c) - f(d)$ in K .

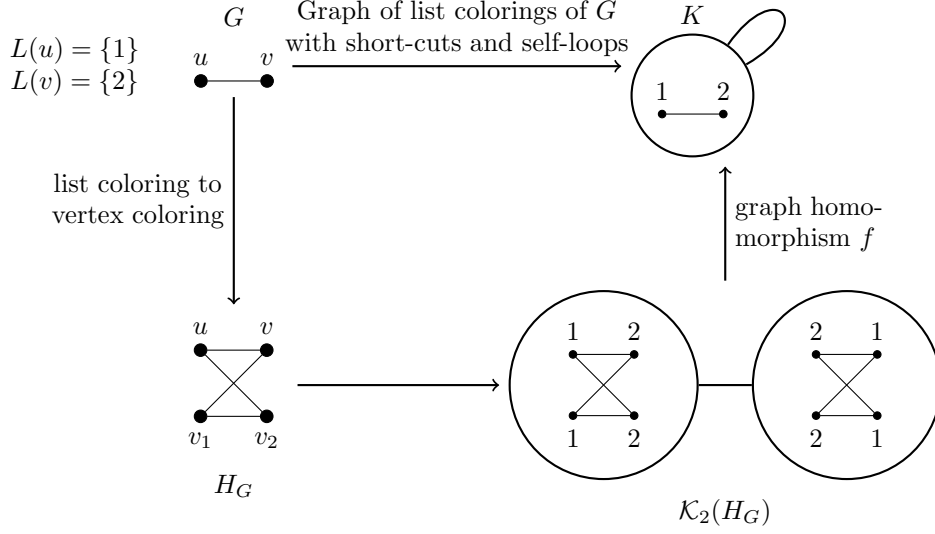


Figure 2: Relations between the graphs G , H_G , K and $\mathcal{K}_p(H_G)$. The choice of G and the available colors determines the other graphs as described in the text. The existence of the graph homomorphism f is established by Lemma 2.

2. The Kempe-exchange κ involves two nodes $u, v \in \{v_1, \dots, v_p\}$. We need to consider the following two subcases. If $H_G(c(u), c(v))$ is connected then $f(c) = f(d)$ and therefore, $f(c) \sim f(d)$, since each node of K has a self-loop. Otherwise, $f(c)$ and $f(d)$ differ with respect to the color classes $c(u)$ and $c(v)$. We show that $f(c)$ and $f(d)$ are connected by a sequence of Kempe-exchanges that swaps the colors in all except a single Kempe-component of $H_G(c(u), c(v))$ and thus $f(c) \sim f(d)$ by the construction of K above. To obtain $f(d)$, we first apply κ to c on H_G and then apply f to the resulting coloring. The Kempe-exchange κ swaps the colors of the connected component of $H_G(c(u), c(v))$ containing u and v , and then f swaps the colors in $H_G(c(u), c(v))$. As a result, $f(d)$ can be obtained from $f(c)$ by swapping the colors in $H_G(c(u), c(v))$ except the one containing u and v in the preimage $f^{-1}(V(G(c(u), c(v))))$.

In summary, for all $c, d \in V(\mathcal{K}_p(H_G))$: $c \sim d$ implies $f(c) \sim f(d)$. \square

The graph homomorphism f induces the equivalence relation \sim_f on $V(\mathcal{K}_p(H_G))$, that is, for $a, b \in V(\mathcal{K}_p(H_G))$: $a \sim_f b$ if $f(a) = f(b)$.

Theorem 2. $\mathcal{K}_p(G)[C_\pi]$ is connected if and only if $\mathcal{K}_p(H_G)$ is connected.

Proof. We noted above that $\mathcal{K}_p(G)[C_\pi]$ is connected if and only if K is connected. Let $f : \mathcal{K}_p(H_G) \rightarrow K$ be the graph homomorphism from Lemma 2.

“Only if” part: Let $\mathcal{K}_p(H_G)$ be connected. Then K is connected since there is a graph homomorphism $\mathcal{K}_p(H_G) \rightarrow K$, and graph homomorphisms preserve connectedness. Therefore, $\mathcal{K}_p(G)[C_\pi]$ is connected.

“If” part: Let $\mathcal{K}_p(G)[C_\pi]$ be connected. Then K is connected. Due to the first isomorphism theorem, $K \cong \mathcal{K}_p(H_G)_{/\sim_f}$ and thus, $\mathcal{K}_p(H_G)_{/\sim_f}$ is also connected. Any two colorings u, v of H_G such that $u \sim_f v$ are connected by Kempe-exchanges since one can be obtained from the other by permuting the colors of the color classes. \square

We show that using the reduction from list to vertex coloring results in a representation of the search space which has a similar diameter compared to the actual search space $\mathcal{K}_p(G)[C_\pi]$.

Theorem 3. $\text{diam}(\mathcal{K}_p(G)[C_\pi]) \leq \lfloor \frac{|V(G)|-1}{2} \rfloor \cdot \text{diam}(\mathcal{K}_p(H_G))$.

Proof. First, note that graph homomorphisms preserve connectedness. Thus, if $\mathcal{K}_p(H_G)$ is connected so is $\mathcal{K}_p(G)[C_\pi]$. Now, for any adjacent nodes $c, d \in \mathcal{K}_p(H_G)$, we count how many Kempe-exchanges are

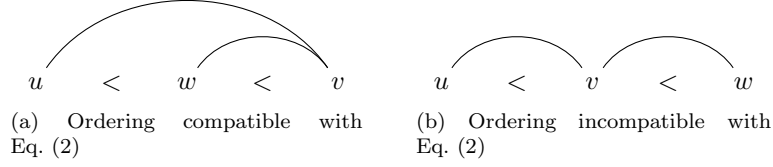


Figure 3: Two vertex orderings of the graph $u - v - w$, $F = \{w\}$.

required to get from $f(c)$ to $f(d)$ in $\mathcal{K}_p(G)[C_\pi]$. Let κ be the Kempe-exchange that is a witness of $c - d$, and let i and j be the involved color classes. If $c \sim_f d$ then, in the worst case, all except one connected component of $G(i, j)$ need to be switched to get from c to d for the reasons stated in cases 1 and 2 in the proof of Lemma 2. There are at most $\lfloor (|V(G)| - 1)/2 \rfloor$ components and at most one Kempe-exchange is required for each of them. If $c \not\sim_f d$ then there is a single Kempe-exchange on G that establishes $f(c) - f(d)$. Thus, a shortest path of maximum length t in $\mathcal{K}_p(H_G)$ corresponds to a path of length at most $t \cdot \lfloor (|V(G)| - 1)/2 \rfloor$ in $\mathcal{K}_p(G)[C_\pi]$. \square

3.2 The connectedness of $\mathcal{K}_p(H_G)$

Given two colorings c and c' of H_G , the algorithm KEMPERECONFIGURATION transforms c into c' as long as there is a sufficient number of colors available. Due to the reduction however, H_G contains a clique on the vertices v_1, \dots, v_p , which implies that $\deg(H_G) \geq p - 1$. Therefore, according to Corollary 1, the clash-free timetables which satisfy timeslot availability requirements are connected if $p > \deg(H_G) \geq p - 1$, that is, $\deg(H_G) = p - 1$. In order to obtain less strict conditions for the connectedness we fix the colors of the clique vertices v_1, \dots, v_p of H_G , and possibly other vertices. As a consequence, we exclude the clique from the recoloring process, so the number of colors required by KEMPERECONFIGURATION is no longer dominated by the clique.

We will first consider the general case, where the colors of some vertices $F \subseteq V(G)$ are assumed to be fixed. We denote by $\overline{F} = V(G) \setminus F$ be the remaining vertices. Further, let $S' \subset S(G)$ be the vertex orderings satisfying

$$\forall u - v - w, u, v \in \overline{F}, w \in F : u < v \Rightarrow w < v. \quad (2)$$

That is, if v is a successor of u and they are adjacent, then all neighbors of v in F must precede v . Fig. 3 shows two examples of vertex orderings of the graph $u - v - w$. For $F = \{w\}$, the ordering shown in Fig. 3a satisfies the condition in Eq. 2 and the one shown in Fig. 3b does not. We will prove next that KEMPERECONFIGURATION does not change the color of any vertex in F if the vertices $V(G)$ are processed according to an ordering in S' . In order to bound the number of colors required for our analysis, we introduce the following generalization of the degeneracy of a graph:

Definition 3 (Subdegeneracy). Let G be a graph and let $F \subseteq V(G)$. The *subdegeneracy* $\text{subdeg}(F, G)$ of G relative to F is defined as:

$$\text{subdeg}(F, G) := \min_{\sigma \in S'} \max_{v \in V(G) \setminus F} \text{pred}(v, \sigma) \quad (3)$$

Note that $\text{subdeg}(F, G) = \deg(G)$ if F is empty and $\text{subdeg}(F, G) \leq \deg(G)$ otherwise. Intuitively, we are looking for a vertex ordering in S' that minimizes the maximum number of adjacent predecessors of any vertex, however, the number of predecessors of any vertex in F is irrelevant.

Remark 1. The ordering constraints in Eq. (2) are reminiscent of the NP-complete problem [15, MS1 and MS2]. However, an ordering $\sigma \in S'$ can be found in polynomial time (if one exists) by a reduction to 2SAT: The reduction adds for each implication in Eq. (2) an appropriate 2SAT clause¹. However, the complexity of determining the subdegeneracy, i.e., the value of the min-max expression in Eq. (3), is an open problem.

Theorem 4. Let c, c' be k -colorings of G that agree on F . Then KEMPERECONFIGURATION returns a sequence of Kempe-exchanges such that

¹We would like to thank Alexander Raß for this observation.

1. all intermediate colorings also agree on F , and
2. no more than $\text{subdeg}(F, G) + 1$ colors are required.

Proof. We first show that the colors of the vertices F are not changed by KEMPERECONFIGURATION. Assume for a contradiction that in some intermediate coloring a vertex $w \in F$ has a color different from $c(w)$. Then w has been recolored because a neighbor u of w preceding it in σ received color $c(w)$. There are two possible reasons: Either u was recolored to $c(w)$ because $c'(u) = c(w)$, but then $c'(w) \neq c(w)$, a contradiction. If this is not the case, then u was recolored in case 1 or 2 of KEMPERECONFIGURATION, because of a neighbor v preceding it. But this is a contradiction to $\sigma \in S'$.

We now show that $\text{subdeg}(F, G) + 1$ colors are sufficient. Since the vertices in F are never recolored, we consider only the vertices \overline{F} . An unused color may be picked for a vertex $v \in \overline{F}$ in case 2 of Algorithm 1. For each $v \in \overline{F}$, there are at most $\text{subdeg}(F, G)$ neighbors of v preceding it, and there are at most $\text{subdeg}(F, G) - 1$ colors different from the color of v present among these vertices. Thus, there is at least one other color available for v . \square

Recall that for any two clash-free timetables we can assume the colors of the clique vertices v_1, \dots, v_p of H_G to be fixed. If we pick $F \subseteq V(H_G)$ such that any two colorings of H_G which agree on the clique also agree on F , then we obtain the following:

Corollary 2. The clash-free timetables that satisfy timeslot availability requirements are connected if $|P| > \text{subdeg}(F, H_G)$.

We return to the general setting and propose a heuristic approach to finding a witness vertex ordering of $\text{subdeg}(F, G)$ for any graph G and $F \subseteq V(G)$. Let $\tilde{S} \subseteq S'$ be the vertex orderings such that the vertices F precede all other vertices. Recall that for any graph G a witness vertex ordering of the degeneracy $\text{deg}(G)$ can be found by repeatedly removing vertices of minimal degree. In a similar fashion, we can determine the value

$$\lambda(F, G) := \min_{\sigma \in \tilde{S}} \max_{v \in V(G) \setminus F} \text{pred}(v, \sigma) .$$

Moreover, this value is equivalently characterized by a max-min expression, analogous to the two characterizations of the degeneracy shown in Eq. (1):

Algorithm 2: VERTEXELIMINATION

input : graph G , vertices $F \subseteq V(G)$
output: ordering $v_1, \dots, v_{|\overline{F}|}$ of the vertices $\overline{F} = V(G) \setminus F$

$G_{|D|} \leftarrow G$
for $i \leftarrow |\overline{F}|$ **downto** 1 **do**
 choose v_i from $\text{argmin}_{v \in \overline{F}} \{d(v, G_i)\}$
 $G_{i-1} \leftarrow G_i - v_i$.
return $v_1, \dots, v_{|\overline{F}|}$

Theorem 5. For any graph G and $F \subseteq V(G)$,

$$\lambda(F, G) = \min_{\sigma \in \tilde{S}} \max_{v \in V(G) \setminus F} \text{pred}(v, \sigma) = \max_{G[F] \subseteq H \subseteq G} \min_{v \in V(H) \setminus F} \{d_H(v)\} .$$

Furthermore, VERTEXELIMINATION produces a witness vertex ordering of $\lambda(F, G)$.

Proof. The proof is based on the remark on the optimality of VERTEXELIMINATION in [25]. Let $\ell = |\overline{F}|$ and for an ordering v_1, \dots, v_ℓ of \overline{F} let $G_i = G[F \cup \{v_1, \dots, v_i\}]$. Further, let

$$\hat{\delta} := \max_{G[F] \subseteq H \subseteq G} \min_{v \in V(H) \setminus F} \{d(v, H)\} .$$

Intuitively, $\hat{\delta}$ is analogous to the degeneracy of G , but the vertices F are irrelevant. If an ordering $\sigma = v_1, \dots, v_\ell$ of \overline{F} is an output of VERTEXELIMINATION then

$$\begin{aligned} \max_{1 \leq i \leq \ell} \text{pred}(v_i, \sigma) &= \max_{1 \leq i \leq \ell} \{d(v_i, G_i)\} \\ &= \max_{1 \leq i \leq \ell} \min_{v \in V(G_i) \setminus F} \{d(v, G_i)\} \leq \hat{\delta} . \end{aligned}$$

The graphs G_i coincide with those in Algorithm 2.

Now let H^* be a graph such that $G[F] \subseteq H^* \subseteq G$ and

$$\min_{v \in V(H^*) \setminus F} \{d(v, H)\} = \hat{\delta} .$$

Let v_1, \dots, v_ℓ be any ordering of \overline{F} and let i be the smallest index such that $H^* \subseteq G_i$. Then v_i must be a vertex of H^* and $d(v_i, G_i) \geq \hat{\delta}$. Therefore, for any ordering v_1, \dots, v_ℓ of \overline{F} , $\max_{1 \leq j \leq \ell} \{d(v_j, G_j)\} \geq \hat{\delta}$, with equality if the vertex ordering is an output of VERTEXELIMINATION. \square

Certainly, the optimality of VERTEXELIMINATION is only established with respect to the subset $\tilde{S} \subseteq S'$. The vertex ordering obtained from the algorithm can potentially be improved by the following post-processing step: Let $v_1, \dots, v_{|\overline{F}|}$ be an output of VERTEXELIMINATION and let k be the largest number such that v_1, \dots, v_k are independent. Then the vertices v_1, \dots, v_k can be moved before the vertices F in the ordering without violating condition (2). The resulting ordering $\sigma' \in S'$ is not in \tilde{S} and can thus not be generated by VERTEXELIMINATION. There is a potential advantage because the construction guarantees that $\max_{v \in \overline{F}} \text{pred}(v, \sigma') \leq \max_{v \in \overline{F}} \text{pred}(v, \sigma)$.

In summary, the heuristic for computing a vertex ordering $\sigma \in S'(G)$ such that the value $\max_{v \in \overline{F}} \text{pred}(v, \sigma)$ is close to $\text{subdeg}(F, G)$ performs the following two steps:

1. Run VERTEXELIMINATION to generate an ordering $v_1, \dots, v_{|\overline{F}|}$ of the vertices \overline{F} .
2. Let $k \in \mathbb{N}$ be the largest number such that v_1, \dots, v_k are independent in G . Move the vertices v_1, \dots, v_k before the vertices F in the ordering.

4 Results

We use the theoretical insights from the previous section to establish the connectedness of clash-free timetables for a range of UTP benchmark instances. Given a conflict graph G , by Corollary 1, the reconfiguration graph of clash-free timetables is connected if $p > \deg(G)$. If timeslot availability requirements are present, we first use the reduction from list to graph coloring described in Section 2.2 to construct the graph H_G , which contains the additional clique v_1, \dots, v_p . We then use the heuristic from the previous section to determine a bound $\text{subdeg}_{ub}(F, H_G) \geq \text{subdeg}(F, H_G)$. The set F of vertices with “fixed” colors contains the clique vertices v_1, \dots, v_p and any other node of G which has only a single available timeslot/color:

$$F = \{v \in V(H_G) \mid |\Gamma(v) \cap \{v_1, \dots, v_p\}| = p - 1\} , \quad (4)$$

where $\Gamma(v)$ denotes the set of vertices adjacent to v . By Corollary 2, the reconfiguration graphs of the clash-free timetables that satisfy timeslot availability requirements are connected if $p > \text{subdeg}(F, H_G)$.

Table 1 indicates the connectedness of the clash-free timetables according to Corollaries 1 and 2 for instances from the CB-CTT, PE-CTT benchmark sets, as well as instances from the University of Erlangen-Nürnberg. All instances are available from the website [13]. The instances **comp01**, ..., **comp21** are from the CB-CTT track of the International Timetabling Competition 2007 (ITC2007) competition. The instances **ITC2_i01**, ..., **ITC2_i24** are from the PE-CTT track of the same competition. The **erlangen** instances are large real-world instances from the engineering department of the University of Erlangen-Nürnberg. The **toy** instance is a small example instance from the website [13]. For each instance we give the number of timeslots p , the degeneracy of the conflict graph $\deg(G)$, and the bound $\text{subdeg}_{ub}(F, H_G) \geq \text{subdeg}(F, H_G)$. Table entries in bold face indicate that the corresponding value $\deg(G)$ or $\text{subdeg}_{ub}(F, H_G)$ certifies the connectedness of the clash-free timetables.

According to the data in Table 1 the clash-free timetables for all CB-CTT and **erlangen** instances are connected, while the conditions imposed by Corollary 1 are not satisfied for any of the PE-CTT

Table 1: For each instance from the CB-CTT, PE-CTT, and Erlangen instance sets, we give the number p of timeslots, $\deg(G)$ and an upper bound $\text{subdeg}_{ub}(F, H_G) \geq \text{subdeg}(F, H_G)$ produced by the heuristic. Values in bold face indicate the connectedness of the clash-free timetables according to Corollaries 1 and 2.

instance	p	$\deg(G)$	$\text{subdeg}_{ub}(F, H_G)$	instance	p	$\deg(G)$	$\text{subdeg}_{ub}(F, H_G)$
comp01	30	23	24	ITC2_i01	45	91	109
comp02	25	23	30	ITC2_i02	45	99	119
comp03	25	22	27	ITC2_i03	45	73	92
comp04	25	17	25	ITC2_i04	45	78	100
comp05	36	26	43	ITC2_i05	45	81	99
comp06	25	17	28	ITC2_i06	45	80	100
comp07	25	20	24	ITC2_i07	45	80	106
comp08	25	20	24	ITC2_i08	45	69	97
comp09	25	22	25	ITC2_i09	45	89	108
comp10	25	18	27	ITC2_i10	45	97	116
comp11	45	27	27	ITC2_i11	45	75	93
comp12	36	22	40	ITC2_i12	45	91	109
comp13	25	17	22	ITC2_i13	45	87	106
comp14	25	17	23	ITC2_i14	45	87	107
comp15	25	22	27	ITC2_i15	45	79	106
comp16	25	18	25	ITC2_i16	45	55	83
comp17	25	17	25	ITC2_i17	45	50	71
comp18	36	14	32	ITC2_i18	45	91	112
comp19	25	23	27	ITC2_i19	45	101	120
comp20	25	19	23	ITC2_i20	45	73	92
comp21	25	23	28	ITC2_i21	45	72	90
erl.2011-2	30	22	32	ITC2_i22	45	98	118
erl.2012-1	30	14	31	ITC2_i23	45	117	128
erl.2012-2	30	20	32	ITC2_i24	45	77	97
erl.2013-1	30	16	30	toy	20	10	11

Table 2: The connectedness of the clash-free timetables for the instances from [22]. For each instance we give the degeneracy $\deg(G)$ of the conflict graph G . Values in bold face indicate the connectedness of clash-free timetables according to Corollary 1.

instance	$\deg(G)$	instance	$\deg(G)$	instance	$\deg(G)$
small_1	54	med_1	59	big_1	60
small_2	41	med_2	67	big_2	68
small_3	98	med_3	67	big_3	64
small_4	69	med_4	69	big_4	80
small_5	84	med_5	87	big_5	75
small_6	24	med_6	101	big_6	93
small_7	68	med_7	120	big_7	111
small_8	84	med_8	98	big_8	82
small_9	124	med_9	121	big_9	77
small_10	136	med_10	64	big_10	77
small_11	34	med_11	97	big_11	76
small_12	22	med_12	78	big_12	76
small_13	146	med_13	105	big_13	84
small_14	100	med_14	92	big_14	74
small_15	79	med_15	101	big_15	127
small_16	118	med_16	145	big_16	115
small_17	120	med_17	126	big_17	184
small_18	60	med_18	188	big_18	131
small_19	141	med_19	173	big_19	159
small_20	28	med_20	153	big_20	144

Table 3: The connectedness of the clash-free timetables for the Metaheuristic Network instances from [31]. For each instance we give the degeneracy $\deg(G)$ of the conflict graph G . Values in bold face indicate the connectedness of clash-free timetables according to Corollary 1.

instance	$\deg(G)$	instance	$\deg(G)$	instance	$\deg(G)$
easy01	15	medium01	49	hard01	68
easy02	19	medium02	53	hard02	67
easy03	13	medium03	52		
easy04	12	medium04	51		
easy05	20	medium05	47		

instances. For eight CB-CTT instances, the upper bound on $\text{subdeg}(F, H_G)$ is sufficient to show that the reconfiguration graphs are connected in the presence of timeslot availability constraints. For the PE-CTT instances, since neither $\deg(G)$ nor $\text{subdeg}_{ub}(F, H_G)$ certifies the connectedness of the reconfiguration graphs, better bounds on $\text{subdeg}(F, H_G)$ are of no use, since $\text{subdeg}(F, H_G) \geq \deg(G)$. Therefore, new techniques are needed for proving the connectedness (or disconnectedness) of the reconfiguration graphs for these instances. A possible reason for this structural difference between the CB-CTT and PE-CTT instances is that the course specification in the former leads to lots of small cliques in the conflict graph, a fact that we will use shortly to determine the subdegeneracy of the conflict graph for the CB-CTT instance `toy`. In contrast, in the PE-CTT problem formulation and also the instances from [22], the event conflicts depend on the students' individual choices, which apparently leads to denser graphs, i.e., graphs with higher degeneracy.

In Tables 2 and 3, the degeneracy values of the corresponding conflict graphs are given for the instance sets from [22] and [31]. On these instances, each timeslot is available for each event. Values in bold face indicate the connectedness of clash-free timetables is established by Corollary 1.

Finally, we will show that for the CB-CTT instance `toy`, the proposed heuristic yields an optimal vertex ordering, i.e., a witness for $\text{subdeg}(F, H_G)$. The instance has in total 20 timeslots and 16 events. In the CB-CTT formulation, the events are grouped into courses. Any two events of a course are conflicting,

Table 4: Instance data of the instance **toy**, available from the website [13].

Course	Events	Conflicts	Unavailable timeslots
TecCos	5	SceCosC, ArcTec, Geotec	8,9,14,15
ArcTec	3	SceCosC, TecCos	16,17,18,19
SceCosC	3	ArcTec, TecCos	–
Geotec	5	TecCos	–

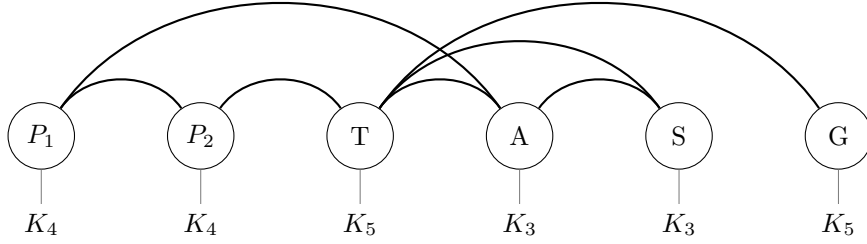


Figure 4: Succinct representation of the graph H_G , where G is the conflict graph of the instance **toy**. All nodes represent cliques as denoted indicated the nodes. Arranging the clique vertices in the shown left-to-right ordering yields a witness of $\text{subdeg}(F, H_G) = 11$.

that is, the events of a course are a clique in the conflict graph. Whenever two courses are conflicting, no two of the corresponding events may be scheduled in the same timeslot. For completeness, the relevant data on events, conflicts and unavailable timeslots is given in Table 4.

Let G be the conflict graph of the instance **toy** and let H_G be the graph that results from the reduction from list to graph coloring. If two courses are in conflict, then the events of both courses are a clique in G . If certain timeslots are unavailable for a course, then the events of the course and then these timeslots form a clique in H_G . Figure 4 shows a succinct representation of the graph H_G . The nodes T , A , S and G correspond to event cliques of the courses TecCos, ArcTec, SceCosC, and Geotec, respectively. The node P_1 represents the timeslots marked unavailable for the course ArcTec and the node P_2 represents the timeslots unavailable for SceCosC. Since no other timeslots are excluded, the corresponding vertices in the graph H_G will not contribute to the subdegeneracy and can be ignored. As a result we get a clique on eight nodes that model the timeslot availability requirements. This clique is divided separated into the two cliques P_1 and P_2 . Two nodes of the shown graph are connected whenever all nodes of the two corresponding cliques are connected.

Let $F = V(P_1) \cup V(P_2)$ and let $\sigma \in S(H_G)$ such that the cliques are arranged in the order P_1, P_2, T, A, S, G with some arbitrary choice of the relative ordering of the vertices within each clique. This ordering is a possible output of the algorithm VERTEXELIMINATION. From

$$\max_{v \in V(H_G) \setminus F} \text{pred}(v, \sigma) = 11 \quad ,$$

we can conclude that $\text{subdeg}(F, H_G) \leq 11$.

Proposition 3. For the instance **toy**, $\text{subdeg}(F, H_G) = 11$.

Proof. Let σ be an ordering of $V(H_G)$ and $V' \subseteq V(H_G) \setminus F$. The maximum number of predecessors adjacent to any vertex of V' in H_G is denoted by

$$p(V', \sigma) = \max_{v \in V'} \text{pred}(v, \sigma) \quad .$$

Note that for a clique $K \in \{T, A, S, G\}$, the value $p(K, \sigma)$ is determined by the last vertex of K in σ . Thus, the value of $p(K, \sigma)$ depends only on the relative order of the last vertices of the cliques $\{T, A, S, G\}$

in σ . Let \tilde{S} be the vertex orderings of H_G such the vertices F precede all other vertices of H_G and let \hat{S} be the total orderings of $\{T, A, S, G\}$. For each ordering $\sigma' \in \hat{S}$ we can pick an ordering $\ell(\sigma')$ of H_G that is compatible with σ' in the sense that the relative ordering of the last vertices of the cliques is in accordance with σ' . We have,

$$\text{subdeg}(F, H_G) = \min_{\sigma \in \tilde{S}} \max_{K \in \{T, A, S, G\}} p(K, \sigma) = \min_{\sigma' \in \hat{S}} \max_{K \in \{T, A, S, G\}} p(K, \ell(\sigma')) .$$

For any ordering $\sigma' \in \hat{S}$ such that $G < T$, we have $p(T, \ell(\sigma')) \geq 13$, because the last vertex of T has at least 13 adjacent predecessors in H_G . Thus, we only need to consider orderings such that $G > T$. Furthermore, since no vertex of G is adjacent to any vertex of A or S , changing the relative order of A and G or S and G does not change the number of adjacent predecessors. Hence, we can assume G is a maximum in any ordering of interest. We enumerate the values of $p(K, \ell(\sigma'))$ all for $K \in \{T, A, S, G\}$ for the 6 permutations of $\{T, A, S\}$:

clique ordering $\sigma' \in \hat{S}$	$p(T, \ell(\sigma'))$	$p(A, \ell(\sigma'))$	$p(S, \ell(\sigma'))$	$p(G, \ell(\sigma'))$
T, A, S, G	8	11	10	9
T, S, A, G	8	14	7	9
A, T, S, G	11	6	10	9
S, T, A, G	11	11	2	9
A, S, T, G	14	6	5	9
S, A, T, G	14	9	2	9

Thus,

$$\text{subdeg}(F, H_G) = \min_{\sigma' \in \hat{S}} \max_{K \in \{T, A, S, G\}} p(K, \ell(\sigma')) = 11$$

□

We can conclude that the proposed heuristic produces a witness of $\text{subdeg}(F, H_G) = 11$ on the instance `toy`.

5 Conclusions

We investigated the connectedness of clash-free timetables with respect to the Kempe-exchange operation. This investigation is related to the connectedness of the search space of timetabling problem instances, which is a desirable property, for example for two-step algorithms using the Kempe-exchange during the optimization step. We include timeslot availability requirements in our analysis and derive improved conditions for the connectedness of clash-free timetables in this setting. For this purpose, we introduced the notion of subdegeneracy, which generalizes the degeneracy of a graph. The complexity of determining the subdegeneracy is an interesting open problem. We further showed that our representation of the search space of clash-free timetables that satisfy timeslot availability requirements is a suitable one with respect to the connectedness properties and the diameter of the search space. Our results indicate the connectedness of the clash-free timetables for a number of benchmark instances.

For future research, other properties of feasible timetables such as overlap-freeness may be considered as well. Furthermore, two kinds of possible improvements may be considered with respect to establishing the connectedness of clash-free timetables in the presence of timeslot availability requirements: Both, a better analysis of Algorithm 1 and a better heuristic approach (or exact algorithm) for determining the subdegeneracy may lead to a lower number of timeslots required to certify the connectedness of clash-free timetables.

References

- [1] Brenda S. Baker and Edward G. Coffman, Jr. Mutual exclusion scheduling. *Theoretical Computer Science*, 162(2):225–243, 1996.

- [2] Vladimir Batagelj and Matja Zavernik. Fast algorithms for determining (generalized) core groups in social networks. *Advances in Data Analysis and Classification*, 5(2):129–145, 2011.
- [3] Marthe Bonamy, Matthew Johnson, Ioannis Lignos, Viresh Patel, and Daniël Paulusma. Reconfiguration graphs for vertex colourings of chordal and chordal bipartite graphs. *Journal of Combinatorial Optimization*, 247(1):1–12, 2014.
- [4] Paul Bonsma. The complexity of rerouting shortest paths. In *Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 222–233, 2012.
- [5] Paul Bonsma and Luis Cereceda. Finding paths between graph colourings: PSPACE-completeness and superpolynomial distances. *Theoretical Computer Science*, 410:5215–5226, 2009.
- [6] Alex Bonutti, Fabio De CESCO, Luca Di Gaspero, and Andrea Schaerf. Benchmarking curriculum-based course timetabling: Formulations, data formats, instances, validation, and results. *Annals of Operations Research*, 194(1):59–70, 2012.
- [7] Michael W. Carter. A comprehensive course timetabling and student scheduling system at the University of Waterloo. In *Selected Papers from the Third International Conference on Practice and Theory of Automated Timetabling III (PATAT)*, pages 64–82, 2001.
- [8] Luis Cereceda, Jan van den Heuvel, and Matthew Johnson. Connectedness of the graph of vertex-colourings. *Discrete Mathematics*, 308(5–6):913 – 919, 2008.
- [9] Dominique de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19(2):151–162, 1985.
- [10] Dominique de Werra. The combinatorics of timetabling. *European Journal of Operational Research*, 96(3):504–513, 1997.
- [11] Dominique de Werra. Restricted coloring models for timetabling. *Discrete Mathematics*, 165–166:161–170, 1997.
- [12] Luca Di Gaspero, Barry McCollum, and Andrea Schaerf. The second international timetabling competition (ITC-2007): Curriculum-based Course Timetabling (Track 3). In *Proceedings of the 1st International Workshop on Scheduling, a Scheduling Competition (SSC)*, 2007.
- [13] Luca Di Gaspero and Andrea Schaerf. Curriculum-based course timetabling web-site. <http://satt.diegm.uniud.it/ctt/>, 2013. Accessed September, 2013.
- [14] Carl Feghali, Matthew Johnson, and Daniël Paulusma. Kempe equivalence of colourings of cubic graphs. *CoRR*, abs/1503.03430, 2015.
- [15] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [16] Parikshit Gopalan, Phokion G. Kolaitis, Elitza Maneva, and Christos H. Papadimitriou. The connectivity of Boolean satisfiability: Computational and structural dichotomies. *SIAM Journal on Computing*, 38(6):2330–2355, 2009.
- [17] Takehiro Ito, Erik D. Demaine, Nicholas J. A. Harvey, Christos H. Papadimitriou, Martha Sideri, Ryuhei Uehara, and Yushi Uno. On the complexity of reconfiguration problems. *Theoretical Computer Science*, 412(12–14):1054–1065, 2011.
- [18] Marcin Kamiński, Paul Medvedev, and Martin Milani. Complexity of independent set reconfigurability problems. *Theoretical Computer Science*, 439:9–15, june 2012.
- [19] Marcin Kamiński, Paul Medvedev, and Martin Milanič. Shortest paths between shortest paths and independent sets. In *Proceedings of the 21st International Workshop on Combinatorial Algorithms (IWOCA)*, pages 56–67, 2011.
- [20] Michel Las Vergnas and Henri Meyniel. Kempe classes and the Hadwiger conjecture. *Journal of Combinatorial Theory, Series B*, 31(1):95–104, 1981.

- [21] Rhydian Lewis. *Metaheuristics for University Course Timetabling*. PhD thesis, Napier University, Edinburgh, Scotland, 2006.
- [22] Rhydian Lewis and Ben Paechter. New “harder” instances for the university course timetabling problem. <http://www.soc.napier.ac.uk/~benp/centre/timetabling/harderinstances.htm>, 2013. Accessed September, 2013.
- [23] Zhipeng Lü and Jin-Kao Hao. Adaptive tabu search for course timetabling. *European Journal of Operational Research*, 200(1):235–244, 2010.
- [24] Giorgio Lucarelli. *Scheduling in Computer and Communication Systems and Generalized Graph Coloring Problems*. PhD thesis, Athens University of Economics and Business, 2009.
- [25] David W. Matula. A min-max theorem for graphs with application to graph coloring. *SIAM Review*, 10(4):467–490, 1968.
- [26] Barry McCollum, Paul McMullan, Edmund K. Burke, and Rong Parkes, Andrew J. and Qu. The second International Timetabling Competition: Examination timetabling track. Technical Report QUB/IEEE/Tech/ITC2007/Exam/v4.0/17, Queen’s University, Belfast, September 2007.
- [27] Bojan Mohar. Kempe equivalence of colorings. In Adrian Bondy, Jean Fonlupt, Jean-Luc Fouquet, Jean-Claude Fournier, and Jorge L. Ramírez Alfonsín, editors, *Graph Theory in Paris*, Trends in Mathematics, pages 287–297. Birkhäuser, 2007.
- [28] Moritz Mühlenthaler and Rolf Wanka. On the connectedness of clash-free timetables. In *Proceedings of the 10th International Conference on the Practice and Theory of Automated Timetabling (PATAT)*, pages 330–346, 2014.
- [29] Nelishia Pillay. A survey of school timetabling research. *Annals of Operations Research*, 218(1):261–293, 2014.
- [30] Olivia Rossi-Doria, Michael Sampels, Mauro Birattari, Marco Chiarandini, Marco Dorigo, Luca M. Gambardella, Joshua Knowles, Max Manfrin, Monaldo Mastrolilli, Ben Paechter, Luis Paquete, and Thomas Stützle. A comparison of the performance of different metaheuristics on the timetabling problem. In Edmund Burke and Patrick Causmaecker, editors, *Practice and Theory of Automated Timetabling IV*, volume 2740 of *Lecture Notes in Computer Science*, pages 329–351. Springer Berlin Heidelberg, 2003.
- [31] Olivia Rossi-Doria, Michael Sampels, Mauro Birattari, Marco Chiarandini, Marco Dorigo, Luca M. Gambardella, Joshua Knowles, Max Manfrin, Monaldo Mastrolilli, Ben Paechter, Luis Paquete, and Thomas Stützle. Supporting material for the paper [30]. <http://iridia.ulb.ac.be/supp/IridiaSupp2002-001/index.htm>, 2014. Accessed February, 2014.
- [32] Andrea Schaerf. A survey of automated timetabling. *Artificial Intelligence Review*, 13(2):87–127, 1999.
- [33] Katja Schimmelpfeng and Stefan Helber. Application of a real-world university-course timetabling model solved by integer programming. *OR Spectrum*, 29(4):783–803, 2007.
- [34] George Szekeres and Herbert S. Wilf. An inequality for the chromatic number of a graph. *Journal of Combinatorial Theory*, 4(1):1–3, 1968.
- [35] Marcin Wrochna. Homomorphism reconfiguration via homotopy. In *32nd International Symposium on Theoretical Aspects of Computer Science, STACS 2015, March 4-7, 2015, Garching, Germany*, pages 730–742, 2015.